110823

31794 P. 8

Production and Quality Assurance Automation in the Goddard Space Flight Center Flight Dynamics Facility*

K. B. Chapman, C. M. Cox, and C. W. Thomas
Computer Sciences Corporation (CSC)
Lanham-Seabrook, Maryland, USA

O. O. Cuevas and R. M. Beckman
Goddard Space Flight Center (GSFC)
Greenbelt, Maryland, USA

ABSTRACT

The Flight Dynamics Facility (FDF) at the National Aeronautics and Space Administration (NASA) Goddard Space Flight Center (GSFC) generates numerous products for NASA-supported spacecraft, including the Tracking and Data Relay Satellites (TDRSs), the Hubble Space Telescope (HST), the Extreme Ultraviolet Explorer (EUVE), and the Space Shuttle. These products include orbit determination data, acquisition data, event scheduling data, and attitude data. In most cases, product generation involves repetitive execution of many programs. The increasing number of missions supported by the FDF has necessitated the use of automated systems to schedule, execute, and quality assure these products. This automation allows the delivery of accurate products in a timely and cost-efficient manner. To be effective, these systems must automate as many repetitive operations as possible and must be flexible enough to meet changing support requirements.

The FDF Orbit Determination Task (ODT) has implemented several systems that automate product generation and quality assurance (QA). These systems include the Orbit Production Automation System (OPAS), the New Enhanced Operations Log (NEOLOG), and the Quality Assurance Automation Software (QA Tool) (Chapman et al., 1993; Chapman et al., 1994). Implementation of these systems has resulted in a significant reduction in required manpower, elimination of shift work and most weekend support, and improved support quality, while incurring minimal development cost.

This paper will present an overview of the concepts used and experiences gained from the implementation of these automation systems.

INTRODUCTION

As part of the FDF, the ODT is responsible for processing tracking data; performing orbit determination; and generating state vectors, ephemeris data, and station contact scheduling products. The ODT makes use of the FDF's two IBM 9121/490 mainframe computers to generate its products. The jobs necessary to generate the products must be set up and executed according to schedules specified by agreements between each mission and the FDF. Jobs are executed either in batch mode using Job Control Language (JCL) or in the foreground. Products are generated daily and must be quality assured and delivered to the appropriate users. These products are used by other groups in the FDF and by outside users for generating acquisition data, spacecraft onboard computer ephemerides, and flight operations and science mission support schedules. The products are necessary for the acquisition of spacecraft by tracking sites, prediction of tracking schedules and

* This work was supported by the National Aeronautics and Space Administration (NASA)/Goddard Space Flight Center (GSFC), Greenbelt, Maryland, under Contract NAS 5-31500.

spacecraft events, and generation of spacecraft computer uploads used in navigation. Errors in the products could result in lost support and science data, missed tracking, or the loss of the spacecraft. Thus, these products and data are extremely important in the day-to-day operations and safety of the supported spacecraft. The standard support provided by the ODT in the GSFC FDF is illustrated in Figure 1.

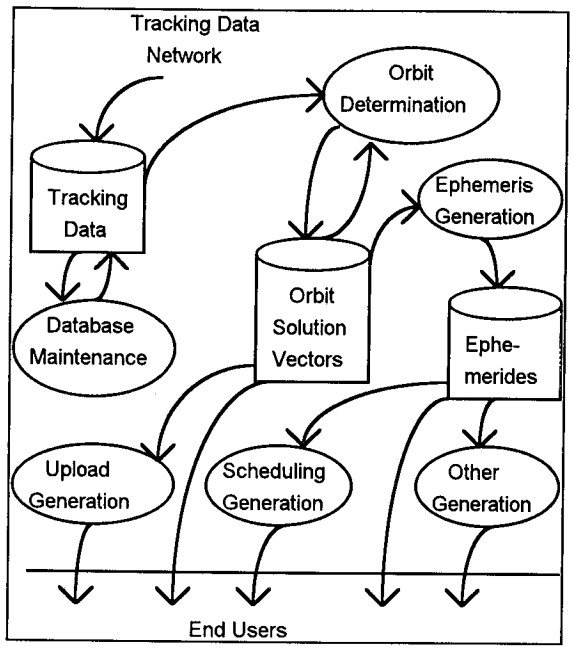

Figure-1: Orbit Determination Task Standard Support

In addition to the standard support, the ODT also performs analysis on the data and products that are generated. The analysis is performed to trend and update QA parameters, to aid in maneuver planning, and to monitor the orbital evolution of the mission. Analysis parameters include the spacecraft's semimajor axis, tracking data statistical information, and the derived coefficients used in the orbit solution. Previously, this type of analysis involved manually transcribing values obtained from job output into required reports.

In the past, the generation, QA, and delivery of products were labor intensive. Users manually edited JCL, changing up to 33 different parameters per job before submitting the JCL. The resulting output and printouts, most containing thousands of lines of output, were hand-checked by the users to perform QA using an average of 60 to 70 parameters per product. Deliveries were performed by relying on a user's knowledge of what product went to what user. Previous to any automation, daily product generation required two to three personnel for 4 to 6 hours a day. The QA process required three to four staff personnel for up to 4 hours per day, and product delivery took two people 2 hours. Thus, the combined production, QA, and delivery processes resulted in up to 38 staff hours per day for nominal support. Not only was this process costly, but, because of the amount of time it took to generate a completed product, delivery schedules were being impacted. Also, the number of required products continued to grow as new missions, often requiring more complex support, were added (see Figure-2).

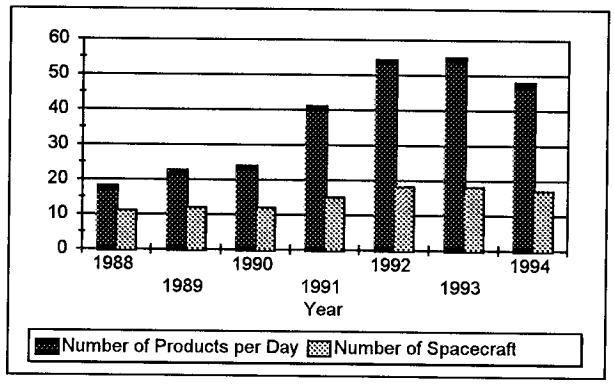

Figure-2: Orbit Determination Task Workload

In order to reduce costs, improve quality, and increase productivity, these manual processes were automated. This paper describes the ODT's product generation processes that required automation, discusses the automation tools generated, summarizes some lessons learned, and presents results and conclusions.

PRODUCTION PROCESS

Analysis of the ODT production cycle defined five product generation processes: scheduling, generation, QA, delivery, and tracking (see Figure-3). Because every ODT product passes through these steps, the emphasis was placed on the definition and execution of the processes for the entire

workload, not just on a product-by-product basis. For example, if there are 50 products in the day's worklist, to schedule, generate, QA, and deliver each product one by one would be costly. Since each process is necessary for the completion of an ODT product, these processes were targeted for automation as a means of reducing the cost of support.

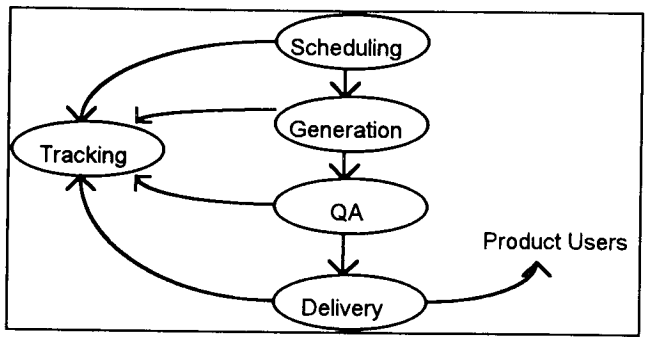

Figure-3: Product Generation Processes

The five product generation processes are decribed in the following subsections.

Product Scheduling

Products are generated according to support schedules determined by mission requirements and customer needs. This process is complicated because the missions have different delivery and support requirements for their products. These are specified in the Interface Control Documents (ICDs) and mission support documentation (for example, GSFC Flight Dynamics Division, 1991), and are determined through extensive analysis of the mission accuracy requirements. The current support includes 93 different product generation runs with varying schedules. Requiring users to remember an involved product schedule increases the risk of incorrect support. This process needs to be flexible enough to accommodate combinations of every possible product schedule (see Table-1). Also, the scheduling is subject to change depending on the status of the spacecraft or the requirements of the customer receiving the product. Scheduling also pertains to the various delivery methods employed after a product was generated. If a product is scheduled for generation, it may also need to be scheduled for the various available deliveries.

Table-1: Example of Schedule Variance

Product	Schedule
EUVE Orbit Solution and Ephemeris	Every Day
HST Orbit Solution and Ephemeris	Every Other Day
UARS 8 Week Ephemeris	Every Thursday
IMP-8 Long Ephemeris	First Friday of Month

Product Generation

Product generation involves submitting the correct software with the correct input to create the end product. The products are generated by a variety of software, such as the Goddard Trajectory Determination System (GTDS) (Bleich, 1994), which is the primary orbit determination and product generation package for the ODT. Missions might have different requirements for similar products. For example, two missions may require TDRS ephemerides with different timespans. In addition, special support is sometimes necessary for product generation, such as following spacecraft maneuvers.

Setting up the product runs involves calculating and inserting proper timespans, orbital elements, force modeling, and other input into the run stream, and submitting it to the system. In many cases, input is required in different locations and formats. For example, a GTDS execution to perform an orbit determination solution, generate an ephemeris, and perform a comparison might need at least three different timespans as input.

Product Quality Assurance

QA is performed to ensure that products are free from anomalies resulting from incorrect input data, corrupt tracking data, environmental events (e.g., solar activity), human error, or spacecraft anomalies. All products are quality assured twice. During initial product generation, ODT personnel perform a preliminary QA on all products by reviewing basic parameters. Then a second group

of ODT personnel perform a detailed QA on the product. Up to 110 parameters from each product are checked against predetermined quality tolerances. Items checked include product data quality (i.e., tracking data statistics, computed or estimated values) and product data consistency (i.e., timespans, correct file names). These data are often spread throughout the output. A subset of the data items used in the QA is recorded in a permanent log to serve as a record and for analysis and trending. The tolerances used are derived from mission requirements, software specifications, and analysis. If a product fails QA, ODT personnel decide if the product should be regenerated with modified input or if the tolerances should be overridden and the product passed for delivery.

Product Delivery

Product deliveries occur in several different ways, and the workload for each delivery type is decided by the products generated and the schedule of deliveries. The delivery of the products consists of copying generated products to operational data files (promotion) and updating a delivery log to inform internal elements that products are ready for their use. It also involves transmitting or delivering products to external sources, such as Payload Operational Control Centers (POCCs) or science centers. Many of the external elements use different methods to receive their products. Transmissions take place over teletype, through Ethernet, or via the NASA Communications network (Nascom). Data may also be received as hardcopy or on a 9-track tape. Deliveries have to be carefully coordinated with each site to ensure that the proper product is delivered in the proper fashion.

Product and Event Tracking

Product and event tracking is a process that occurs throughout the entire production cycle, to satisfy the requirement to maintain a record of activities performed by both the system and the users. Such records should maintain a running account of the jobs that have been run, the products that have been generated and delivered, any anomalies that might have occurred, special requests, and shift turnover. This process is also used to maintain key statistics and QA parameters for future analysis.

In the past, these logs were kept as handwritten or typed manual logs in many groups of the FDF. Problems with the old paper system included missing and illegible entries and the need to consult multiple logs to gain information. Also, with a paper log, only one person could efficiently read and write to it at a time, and that person must be at the same physical location as the log.

AUTOMATION OF THE PRODUCTION PROCESSES

ODT product generation activities were automated by developing several system utilities, which were created as another layer over the existing systems in use (see Figure-4). This was done because the institutional product generation software already existed, and it would have been too expensive to modify it. The systems need to handle the wide range of different product generation programs, and should be able to accommodate new programs without modification. Creating the automation separately was a cost-effective means of implementing improvements as soon as the pieces were ready. Because the generation programs execute primarily in batch mode with JCL, the automation systems deal primarily with configuring the JCL and input data to properly generate and deliver products. A menu system ties the automation systems together under a single user interface (UI).

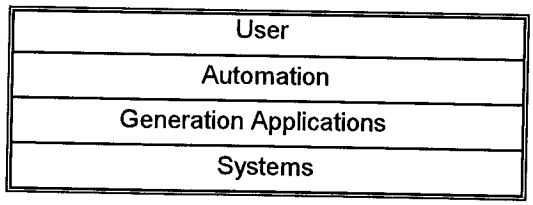

Figure-4: Relationship of Automation Layer to Applications and System

To handle ODT support variability (support schedules, timespans, satellite names, etc.), input configuration files were used to avoid the need for major system updates. Hardcoded parameters were avoided so a change in support would not

necessitate a change in the components of the automation system as well.

The automation also had to accommodate nonstandard or anomalous support. While the ultimate automation would be a total "hands-off" system, there are cases where control of the process should be returned to the user. In the ODT's case, the capability for manual intervention at key points in a process was all that was necessary. Requirements for this capability were a function of the type of support, the environment, the expected frequency of nonstandard support, and the potential impact if operations were delayed.

With the large number of jobs submitted on a regular basis, the users and system needed a means of determining whether processes have been completed. This information is required for system error detection and correction and process logging. Process status information was also useful for notifying and executing subsequent processes. Process status traceability was accomplished through log files and status file updates.

The ODT first developed the OPAS to automate the scheduling and product generation processes. Next, the delivery process was automated with the Delivery Tool. UI improvements were then made by implementing panel-driven menus and then developing the QA Tool. Each implementation resulted in further reduction in the time needed to complete a product (see Figure-5).

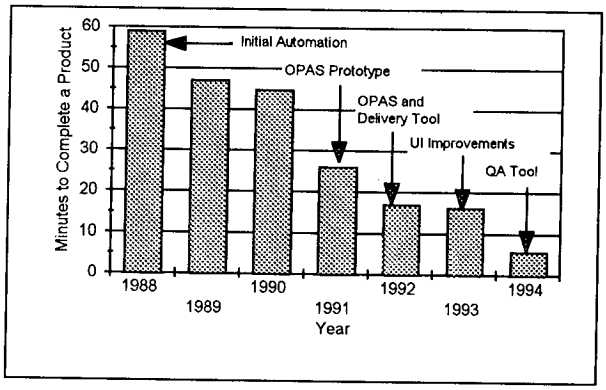

Figure-5: Implementation Dates and Effects on Average Completion Time

All of the automation utilities were developed with significant user input, especially with regard to UIs. Because of the close ties between the users and developers, the system closely reflected the user's needs.

The automation utilities for ODT product scheduling and generation, delivery, QA, and tracking are described in the following subsections.

Product Scheduling and Generation—OPAS

OPAS automates the scheduling and product generation. An original attempt at automation was implemented, refined as a newer prototype system, and then implemented as the final system now in place. OPAS makes use of a master requirements file to describe when a job is to be run, provide the updates needed for the runstream execution, and control the delivery processes. When OPAS is executed, its scheduler function creates a status file containing the list of the day's work and the status of its completion (see Figure-6). The status file becomes the link to the other sections of the automation. The OPAS generation function then sets up the jobs specified in the status file in accordance with the information in the requirements file, including date and timespan calculation. The user has the option to edit the completed runstream before execution, to aid in nonstandard support. Frequently, subsequent product runstreams may require input used from a previous setup. To support this, OPAS uses a current data file to store input needed for several jobs, which reduces the amount of user input required. Input that may be required from the previous day is stored in an a priori file. As the jobs are set up and submitted, OPAS updates its status file to indicate that the step has been completed for that product. The updated status file then serves as the notification to subsequent processes that a product is ready for the next step, such as QA or delivery. Also, because manual user setup is still available, anomalies can be easily worked around without the services of the maintenance personnel.

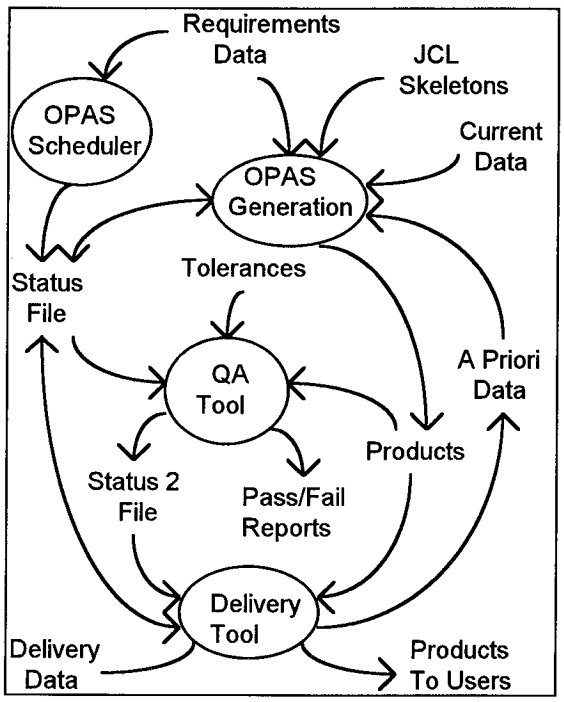

Figure-6: OPAS Functional Data Flow

Product Delivery—OPAS Delivery Tool

The ODT implemented the Delivery Tool function of OPAS to help automate the delivery processes. When executed, the Delivery Tool checks the OPAS status file for the list of the day's work for the type of delivery selected by the user (see Figure-6). It also checks the status file to see if the prerequisite steps have all been completed. The user can then instruct the system to deliver all of the products for that type or individual products. The Delivery Tool also updates the status file to indicate that delivery processes have been performed to maintain accountability. All of the UIs for the Delivery Tool functions operate in the same way where possible and allow for delivery of products that may have been generated but were not in the schedule. Information that aids in the delivery of products, such as file names and product destinations, is stored in delivery data files that are input to the Delivery Tool. The files can be easily modified to fit support requirements.

Product Quality Assurance—QA Tool

Automation of QA required that the data items to be checked be extracted from the output of the product generation phase, checked, and reported. Because a variety of software is used to generate the products, the system could not be coded for the output of any single product. It had to be flexible and generic, with the specified data items and their locations user specified. The tolerances and the operations (i.e., =, <>, <, >, etc.) required in the process also had to be user specified.

The QA Tool is currently implemented as a prototype. The software runs instream with the product generation at the end of the batch run. It extracts user-specified data items from the product output and checks the values against user-specified tolerances (see Figure-6). Depending on the results of the tolerance checking, a flag for each data item is set to pass or fail. Reports are generated to inform the user of the results, and these take the place of the manual logging of data items for recordkeeping and analysis. More data are now available for analysis and recordkeeping. A UI allows the user to quickly ascertain the results of a particular product generation or of the entire day's work. The UI makes use of the OPAS status file, creating an updated version that indicates the pass/fail status of each product. Changes to the production software necessitates, at most, a configuration file change in the QA Tool, not a software update. *Because the user specifies in a single central location the desired data items, their locations, and the tolerances to use, the output from any existing or new software can be checked.*

Product Tracking—NEOLOG

NEOLOG is an online database implementation of the activities log that complements the accountability and tracking provided by OPAS. It allows entries to be made under several different categories and allows entries to be made from runstreams automatically or from interactive sessions with a user. Any user can access the log from any terminal, and multiple users can access the log simultaneously. All production and delivery runs in the ODT write information into the log, as do the analysts performing the work. The end result is a long-term running record of activities

and job executions that can be used for troubleshooting, analysis, and activities tracking. Typically, a log file contains up to a year's worth of entries, and previous years are easily accessible.

LESSONS LEARNED

Significant lessons have been learned from the use and implementation of product generation automation in the GSFC FDF. A key concept is the importance of analyzing the procedures involved in a process to identify repetitive and redundant user actions. Sometimes gains in efficiency are realized through simple procedural changes. Reducing and simplifying procedures also has the benefit of reducing the size of the automation. Other key lessons involve the areas of UIs, reliability, training, and requirements definition.

User Interfaces

The use of UIs to control the system requires special consideration. It is important to keep the interfaces as consistent as possible so that similar functions require similar user actions. Also key is keeping UIs logically organized and easy to use and understand; the urge to create overdone UIs should be firmly resisted. This significantly speeds user familiarization, makes the process more efficient, and reduces the chances for erroneous input. Also, UIs for individual utilities in the automation should be configurable or have the capability to be bypassed. This offers a high degree of flexibility in combining processes and eliminating the need for user input.

Reliability

Reliability is characterized by system robustness, accuracy, and ease of maintenance. The best method for achieving reliability is to keep the system simple. Thorough testing prior to implementation should be conducted to ensure robustness and accuracy. All of the systems implemented by the ODT went through thorough independent testing. By making control and data parameters configurable, maintenance is limited to file and parameter updates. Sufficient configuration management should be in place to ensure that configurations are correct, changes are traceable, and quality controls are enforced. However, *the configuration management must not stifle quick and effective responses to problems.* In the ODT, configuration management of the automation systems is handled by personnel who also participate in the generation of products. Use of the system results in a familiarity that enhances the quick responses for changing requirements. The amount of software maintenance has been reduced significantly by the fact that most changes are now simple configuration file updates instead of coding changes. To avoid any impact that might arise on "off" days due to flawed maintenance, updates are discouraged on Fridays or any day before a holiday.

Training

For the ODT, training issues can be broken into two categories: system training and product familiarity. System training for an automation system is the same as with any other system. The users must be trained in the availability and use of the automation system's capabilities. Again, keeping the functionality of utilities and user interfaces consistent can reduce the time it takes to train users. In the case of automation, the usual resistance and mistrust of a new system by users may be heightened by the fact that many processes now occur out of view. Training and testing help, but if the system is designed to allow manual intervention as a backup, some of the resistance can be alleviated.

As processes and QA become more automated, the user becomes less involved in creating the product. This may result in reducted familiarity with the products and the generation software being used. In the FDF, this is a concern because the support for maneuvers and missions still involves a lot of manual work and analysis, requiring an in-depth knowledge of the products and support software.

Reducing automation to keep users familiar with the software and products is essentially the same as subsidizing the training budget through increased production costs. It is preferable to address the

issue with ongoing training, instead of reducing the amount of automation for production. Graphic feedback from the system may also help, as long as it does not unnecessarily add to the completion time for a product. This means that training costs and issues must be specifically addressed as efficiency is gained through automation. In the case of the ODT, familiarity with the products is maintained through analysis and special requests, as well as other training exercises. In fact, the automation is now freeing up time to perform more analysis, which improves the quality of support.

Requirements Definition

When drafting requirements for new product generation software, special consideration should be given to defining the parts of the output that truly define the quality of the product. While all of the output may be required as a product or for detailed analysis, usually smaller portions (that may be scattered throughout the output) are needed as a "quick look" to indicate the quality of a product. This information could then be provided as a condensed report that is easier to check and incorporate into other utilities. This requires that attention be paid to the potential uses and users of a particular system early in its development.

RESULTS AND CONCLUSIONS

After implementing the automation software, the ODT found that to create an effective automation system, attention must be paid to the reliability of the automation, to the training required to execute and maintain the system, to product familiarity, and to the design of software maintenance releases of product genrating systems. By implementing the automation system, ODT personnel were able to make their product generation and QA more efficient (see Figure-7). Product generation time was reduced to 2 staff hours a day. QA time was reduced from an average of 12 staff hours a day to 1 to 2 staff hours, and delivery was reduced to 1 staff hour. Implementation of the automation systems allowed the FDF to provide operational phase orbit determination and navigation support more effectively for more missions, without having to significantly increase staff or make expensive changes to product generation systems.

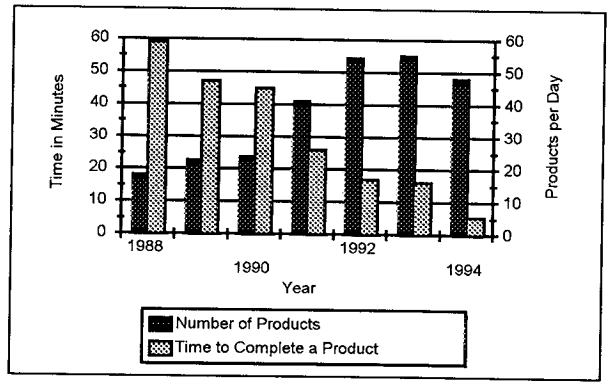

Figure-7: Workload Versus Average Product Completion Time

ACKNOWLEDGMENT

The authors acknowledge Mark Schmitt of Computer Sciences Corporation for his significant contributions to the early develoment of OPAS.

REFERENCES

Chapman, K., Thomas, C., and Schmitt, M. (September 1993). *Orbit Production Automation System Product Generator (OPASPG) System Description and User's Guide* (Report 553-FDD-92/017R0UD0). Goddard Space Flight Center, Flight Dynamics Division, Greenbelt, MD.

Chapman, K. and Thomas, C. (March 1994). *Quality Assurance Automation Software (QA Tool) Software Requirements Specification* (Report 553-FDD-93/097R0UD0). Goddard Space Flight Center, Flight Dynamics Division, Greenbelt, MD.

Goddard Space Flight Center, Flight Dynamics Division (October 1991). *Flight Dynamics Division (FDD) Generic Data Product Formats Interface Control Document* (Report 553-FDD-91/028). Goddard Space Flight Center, Flight Dynamics Division, Greenbelt, MD.

Bleich, A. (February 1994). *Goddard Trajectory Determination System (GTDS) User's Guide, Revision 2, Update 7* (Report CSC/SD-85/6738R2UD7). Computer Sciences Corporation, Lanham-Seabrook, Maryland.